Exigences en ingénierie systèmes basée modèles

Patrice Micouin

Exigences en ingénierie systèmes basée modèles

PATRICE MICOUIN

LABORATOIRE DES SCIENCES DE L'INFORMATION ET DES SYSTEMES,
ARTS ET METIERS PARIS'TECH,
2, COURS DES ARTS ET METIERS, 13100 AIX-EN-PROVENCE, FRANCE
PATRICE.MICOUIN@INCOSE.ORG

***Résumé* - Après un rappel du type de systèmes considérés et d'un processus de conception système en phase avec l'état de l'art, nous décrivons ce que nous entendons par modèles et langage de modélisation. Nous rappelons ensuite les principaux éléments d'une théorie des exigences fondée sur le concept de propriété (PBR) compatible d'une approche d'ingénierie système basée sur des modèles (MBSE). Nous montrons ensuite comment définir, dans un langage de modélisation, des exigences basées sur le concept de propriété (PBRs) au sein même d'un modèle de conception de système, d'appliquer à ce modèle en développement un processus de conception système conforme à l'état de l'art. Nous pensons ainsi proposer des processus de définition des exigences et de solutions des systèmes qui renouvellent les pratiques actuelles du MBSE et les rendent plus adaptées au développement des systèmes d'aujourd'hui et de demain.**

***Abstract* – After a reminder about systems under consideration and a system design process consistent with the state of art, we describe what we mean by models and modeling languages. We remind then the main elements of a property based requirement (PBR) theory in line with a model based system engineering (MBSE) approach. Then, we show how to define PBRs inside system design models, thanks to a modeling language and how to perform a system design process in line with the state of art. Thus, we consider that our proposals about requirement and solution definition processes will improve MBSE current practices and make them more suitable for development of today and future systems.**

***Mots clés* – Système, Propriété, Modèle, Exigence, Conception.**
***Keywords* – System, Property, Model, Requirement, Design.**

1 INTRODUCTION

Hormis les échanges verbaux, les documents (quel qu'en soit le support : papier, fichiers ou bases de données électroniques) constituent le principal moyen de communication et de partage de l'information entre les différentes parties-prenantes d'un projet de développement d'un système technologique, ainsi que le principal moyen d'intégration de l'activité de conception avant la réalisation et l'intégration physique du système lui-même.

Cette ingénierie basée sur des documents a montré son efficacité mais aussi des limites. La complexité croissante des systèmes ne peut qu'aiguiser cette contradiction entre d'une part la difficulté à développer des systèmes technologiques toujours plus sophistiqués avec des équipes de plus en plus importantes, réparties géographiquement sur des aires culturelles variées et d'autre part les moyens disponibles en matière d'ingénierie basée sur des documents.

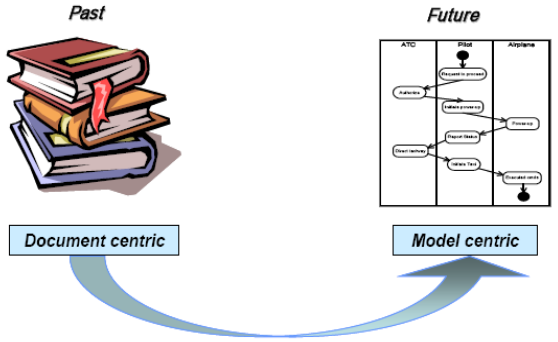

Figure 1. Vision INCOSE du futur de l'ingénierie système.

C'est la raison pour laquelle l'INCOSE (International Council of Systems Engineering) a développé une vision à 2020 de l'ingénierie des systèmes [INCOSE, 2007] dans laquelle le Concile considère que le futur de l'ingénierie des systèmes sera basé sur les modèles et qu'il ouvrira à de nouvelles pratiques d'ingénierie. Aussi le Concile engage-t-il la communauté des ingénieurs et des chercheurs à approfondir leur réflexion sur l'ingénierie des systèmes basée sur les modèles (MBSE) dont il pense qu'elle ne pourra pas être une simple extrapolation non dirigée des pratiques actuelles du MBSE. Le présent article se veut une contribution à cet effort de définition d'une ingénierie des systèmes basée sur les modèles répondant aux attentes exprimées.

Après un bref rappel relatif au type de systèmes considérés et du processus de conception système, nous décrirons ce que nous entendons par modèles. Nous rappellerons ensuite les principaux éléments d'une théorie des exigences fondée sur le concept de propriété (PBRs) cohérente avec une approche du type MBSE. Après quoi, nous monterons comment il est possible de :

1. définir, dans le cadre d'un langage de modélisation, des PBRs au sein même de modèles de conception de systèmes,
2. d'appliquer à ces modèles en développement un processus de conception système conforme à l'état de l'art.

Ce faisant, nous pensons proposer des processus de définition des PBRs et de définition des solutions logique et physique des systèmes qui renouvellent les pratiques actuelles du MBSE et les rendent plus adaptées aux contextes de développement des systèmes d'aujourd'hui et de demain.

2 SYSTEMES

Les systèmes considérés dans le présent article sont, comme l'indique l'annexe D de la norme ISO 15288 [ISO15288, 2002], des systèmes « d'origine humaine, crées et utilisés pour fournir des services dans un environnement défini au bénéfice d'utilisateurs et d'autres parties prenantes ». Ces systèmes technologiques forment ainsi une catégorie particulière de systèmes tels que définis dans leur généralité par l'épistémologue canadien Mario Augusto Bunge dans sa théorie générale des systèmes [Bunge, 1979].

2.1 Les systèmes concrets et propriétés matérielles

Dans [Bunge, 1979], un système est défini comme un objet composite (concret ou abstrait) caractérisé par :

3. sa composition C : collection de ses parties,
4. son environnement E : collection d'entités avec lesquels des parties de C sont en relation,
5. sa structure S : collection des relations que les éléments de C entretiennent entre eux (endostructure) et des relations que des éléments de C entretiennent avec les éléments de E (exostructure).

Ainsi, un système de baro-altimétrie répond à une telle définition dont il est facile d'identifier des caractéristiques possibles:

1. sa composition C : sous systèmes d'acquisition de la pression statique, de conversion analogique/numérique, de calcul de l'altitude barométrique en fonction de la pression et de visualisation de l'altitude.
2. son environnement E : constitué notamment de la masse d'air dans laquelle l'aéronef évolue, du ou des pilotes,
3. sa structure S : constituée par l'ensemble liaisons internes entre composants du système et externes entre composants du système et son environnement. Ces liaisons peuvent être de natures diverses : physique (mécanique, électrique, électromagnétique, etc.), chimique mais aussi psycho sociale pour les humains de l'environnement.

Toujours selon Bunge, tout objet concret est soit un système soit une partie d'un système. Ainsi, les composants d'un système de baro-altimétrie sont à leur tour des systèmes au sens défini précédemment et peuvent être organisés niveau par niveau tandis que le système de baro-altimétrie est lui-même un composant d'un système aéronef lui-même composant d'un sur-système incluant d'autres aéronefs circulant dans un même espace aérien, le système espace aérien. Ce dernier est, en l'occurrence, un système de systèmes [Luzeaux, 2008].

Un objet concret est porteur de propriétés matérielles. Ces propriétés peuvent être catégorisées de différentes façons, notamment en distinguant des:

- propriétés essentielles par opposition à des propriétés accidentelles. Les propriétés d'un objet O qui nous permettent d'inférer que O est du type T sont dites essentielles au type T: fournir une indication de la position verticale est une propriété essentielle d'un système de baro-altimétrie, tandis que l'utilisation d'un écran LCD pour son affichage n'est qu'accidentelle.
- propriétés structurelles par opposition à des propriétés comportementales : disposer d'un écran LCD pour la visualisation est une propriété structurelle tandis que fournir une indication de la position verticale est une propriété comportementale.

La théorie générale des systèmes de Bunge rend compte également des systèmes symboliques (incluant les modèles), et des systèmes abstraits (incluant les systèmes de connaissances) ainsi que des relations qu'ils entretiennent entre eux et avec les systèmes concrets (incluant les systèmes technologiques).

2.2 Les systèmes selon l'EIA 632

Le standard [EIA632, 2003] est un standard conçu pour conduire le développement de systèmes technologiques. Il propose essentiellement deux choses : (1) un cadre conceptuel qui permet de décrire ce qu'est un système technologique et (2) un ensemble de processus qui permettent de conduire l'ingénierie d'un système tel que défini précédemment.

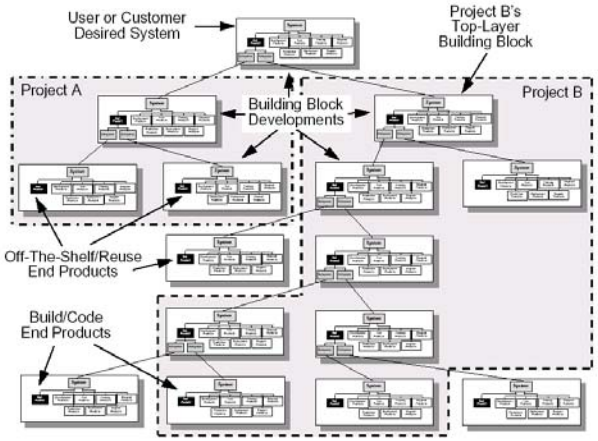

Figure 2. Arbre Système

Nous traiterons du premier point ici tandis que nous traiterons du second au paragraphe suivant.

Selon l'EIA 632, un système se présente comme un arbre constitué d'un certain nombre de blocs de construction (building blocks) comme illustré sur la figure ci-dessous.

Cette structure de bloc, présentée sur la figure 3, décrit un système comme étant constitué de produits de deux sortes : les produits opérationnels (ou finals) qui réalisent les fonctions opérationnelles du système et les produits support (enabling) qui permettent d'assurer les fonctions de support du système (les processus du cycle de vie du système).

Chaque produit (final ou support) peut être soit directement construit/codé/pris sur étagère, soit faire l'objet, à son tour, d'une conception en sous systèmes qui donneront naissance à de nouveaux blocs de construction.

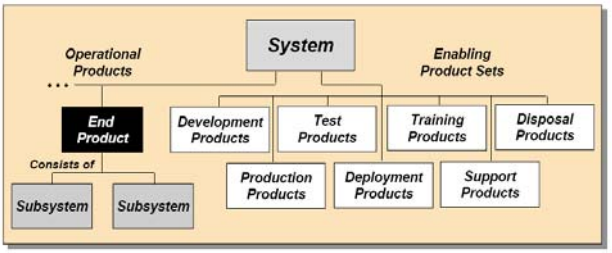

Figure 3. Bloc de construction

On pourrait croire que dans cette vision, les processus et/ou les humains sont oubliés. En fait, cette appréciation est inexacte dans la mesure où les produits de support correspondent aux moyens avec lesquels les operateurs auront à conduire les processus du cycle de vie du système, comme par exemple, un simulateur d'entrainement, un manuel de vol ou un manuel de maintenance, dédiés à la formation et à l'assistance du pilote ou du mécanicien. Le parti pris de ne représenter que les produits ne signifie donc pas que les processus et les humains soient absents de la conception d'un système.

3 PROCESSUS D'INGENIERIE DE SYSTEMES

Les processus d'ingénierie sont aujourd'hui bien connus. Un standard comme l'EIA 632 propose un cadre de développement de systèmes technologiques en définissant les processus à conduire. Il identifie treize processus à appliquer sur chaque bloc de construction d'un système. Compte tenu des limites et des objectifs de cet article nous nous focaliserons

sur deux d'entre eux qui en constituent le cœur : la conception système. Cependant la figure 4 ci-dessous nous donne à voir l'ensemble de ces processus.

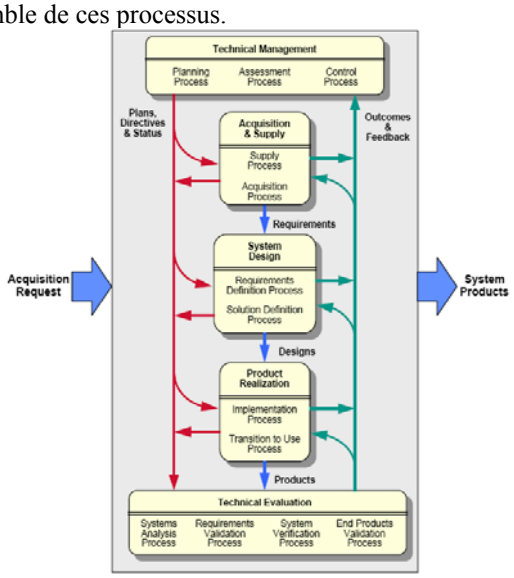

Figure 4. Processus pour l'ingénierie d'un système

3.1 La conception système selon l'EIA-632

La conception d'un bloc de construction d'un système technologique comporte deux phases :

- la définition des exigences qui consiste à faire la synthèse des exigences du système c'est-à-dire celles auxquelles devra se conformer le système à développer,
- la définition d'une solution qui consiste à concevoir deux types de représentations, des représentations logiques et physiques du système à développer en allouant aux différentes entités de ces représentations des exigences dérivées des exigences système. Ces exigences dérivées dépendent essentiellement des choix de conception.

Les deux phases du processus de conception avec les activités et produits associés sont représentées sur la figure 5 ci-dessous.

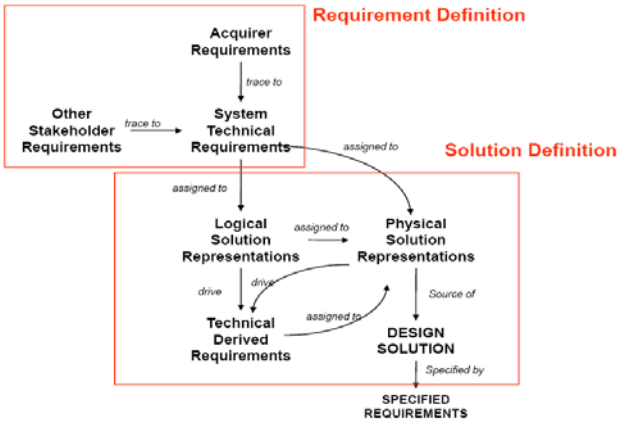

Figure 5. Conception d'un bloc de construction

Un des modes possibles de représentation logique est la modélisation fonctionnelle qui permet de définir des chaînes fonctionnelles articulées les unes aux autres en une architecture fonctionnelle. Cette architecture peut être également associée à des représentations du type diagrammes d'états pour représenter les modes et états du système.

Quand de tels modèles de solution logique ont été identifiés, il devient possible d'allouer certaines exigences (comportementales, plus particulièrement) du système à ces modèles et de les transformer en exigences dérivées allouées à différents éléments de la solution.

La figure 6, ci-dessous, donne un exemple sommaire de conception logique possible d'une exigence comportementale d'une avionique : « Fournir la position verticale de l'aéronef »

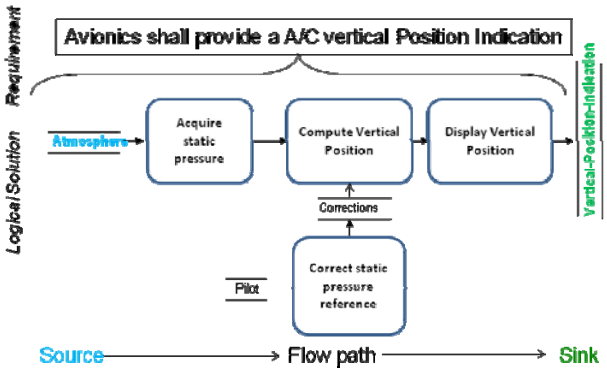

Figure 6. Conception d'une chaîne fonctionnelle

Une fois imaginé(s) le ou les modèles de solution logique, l'EIA 632 propose de définir un ou des modèles de solution physique. Les éléments de solution logique sont alloués à des éléments de solution physique qu'ils matérialisent. Les exigences dérivées allouées aux éléments de solution logique sont alors affectés aux éléments physiques qui les matérialisent. La figure 7, ci-dessous, en donne un exemple sommaire. Certaines exigences du système, parce qu'elles n'étaient pas pertinentes au niveau logique (par exemple, des exigences de poids, de forme ou de volume) sont directement allouées aux modèles physiques et dérivées en exigences allouées à des éléments de ces modèles physiques.

Le résultat de ces processus de conception et de dérivation/allocation des exigences constitue une solution de conception. Une solution de conception correspond à différents ensembles d'exigences spécifiées. Certains de ces ensembles d'exigences peuvent être utilisés pour fabriquer ou coder des éléments constitutifs de la solution, d'autres pour acquérir de tels éléments et d'autres enfin permettent de poursuivre le développement des produits les plus compliqués en sous systèmes.

Cette vision de la conception système est cohérente avec l'architecture générale des systèmes techniques selon l'EIA 632, puisqu'il peut être répété de façon systématique pour chaque bloc de construction.

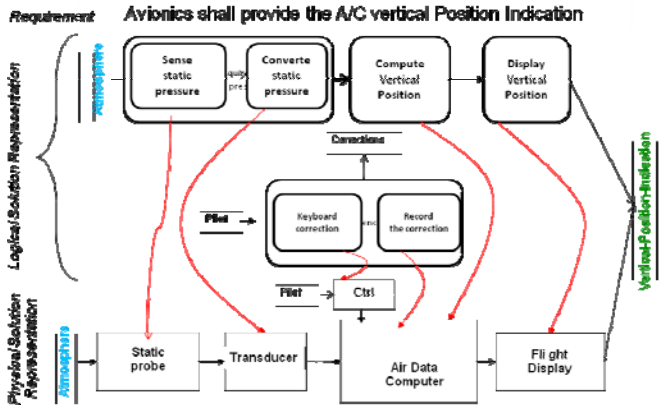

Figure 7. Conception physique d'une chaîne

On notera que dans les deux cas, (figures 6 et7), la composition, l'environnement et la structure du système sont représentées.

3.2 Ingénierie basée documents vs Ingénierie basée modèles

L'ingénierie des systèmes telle qu'elle se pratique le plus souvent et telle qu'elle se présente dans la plupart des standards est une ingénierie basée sur des documents. Cela

signifie que nombre de processus d'ingénierie ont pour condition d'activation la disponibilité d'un ensemble de documents comme par exemple un concept opérationnel et de maintenance et ont pour résultat la délivrance d'un ou plusieurs documents : plans, spécifications techniques de système, descriptions de conception, procédures d'essais, résumés d'accomplissement, etc. Toutefois, cette ingénierie basée sur des documents laisse prise à des incompréhensions ou malentendus entre parties prenantes et à des défauts qui peuvent n'être découverts qu'une fois la réalisation effectuée et entrainer des efforts importants de correction et d'ajustement.

Comme réponse à ces défauts associés à une approche basée sur des documents, l'INCOSE promeut la définition d'une approche basée modèles MBSE qui ne soit pas une simple extrapolation non dirigée des pratiques actuelles du MBSE si elle veut fournir les services attendus (souligné par nous).

Pour y contribuer, nous décrivons maintenant ce que nous entendons par modèles et langages de modélisation, rappelons les principaux éléments d'une théorie des exigences fondée sur le concept de propriété (PBR) de système en phase avec une approche MBSE et nous montrons comment il est possible de définir, dans un langage de modélisation, des PBRs au sein même d'un modèle de conception de système, d'appliquer à ce modèle en développement un processus de conception système conforme à l'état de l'art et de dériver les exigences initiales du système pour les allouer progressivement aux différentes entités du modèle.

4 MODELES

On le sait, il existe une très grande variété de modèles destinés à représenter un système concret. Cette diversité va des maquettes physiques aux systèmes d'équations mathématiques. Dans ce qui suit, l'intérêt est porté sur des représentations d'objets concrets à l'aide de systèmes de symboles.

De tels modèles sont des objets caractérisés par une composition, une structure et un environnement dotés des spécificités suivantes :

1. La composition est une collection de symboles (écrits, dessinés ou prononcés).
2. Ces symboles sont articulés les uns aux autres de manière à produire des énoncés par le biais de règles syntaxiques (endostructure). Ces énoncés peuvent eux-mêmes se recomposer toujours à l'aide de règles syntaxiques de manière à fournir des énoncés de plus en plus élaborés. Les règles syntaxiques garantissent que le modèle est bien formé.
3. L'environnement E du modèle est constitué d'une part du système concret référent et d'autre part de corps de connaissances. Certaines entités, états, événements ou processus du système représenté sont attachés des symboles et des énoncés du modèle soit directement par des relations de dénotation soit par la médiation de concepts et de propositions inscrits dans des corps de connaissances. On a alors la triangulation suivante : les symboles désignent des constructions abstraites (concepts et propositions) qui font référence à des entités ou des propriétés du système concret dénotés par ces symboles. Ces mises en relation constituent l'exostructure du système symbolique qui confère au modèle ce qu'il est convenu d'appeler sa sémantique.

Si le respect de la syntaxe, endostructure du modèle, permet d'assurer que celui-ci est bien formé, son exostructure permet d'affirmer que celui-ci est informé (la question de savoir s'il est bien informé correspond à celle de sa signification et de sa vérité factuelle par rapport au système concret représenté).

Par exemple, un programme logiciel, écrit dans un langage de programmation et syntaxiquement correct, est un modèle du comportement d'un calculateur qui exécuterait le code exécutable correspondant à ce programme, tout comme une partition musicale peut être considérée comme un modèle d'une symphonie exécutée par un orchestre. De même, si on considère la description d'un composant matériel complexe dans un langage de description et syntaxiquement correct, il s'agit d'un modèle de ce composant tel qu'il pourrait être synthétisé et fabriqué à partir de cette description.

On peut noter que cette caractérisation des modèles n'écarte pas les descriptions en langue naturelle. De telles descriptions en langue naturelle (écrite ou orale) constituent également des modèles. Pour pouvoir les mettre hors du champ de l'ingénierie basée modèles (MBSE) il faut donc préciser le type de langages utilisables dans ce cadre. En l'occurrence, les langages adaptés au MBSE sont des langages artificiels, textuels ou graphiques, formels ou semi-formels, qui, d'une part, permettent une analyse syntaxique de manière à garantir que les modèles sont bien formés et d'autre part évitent la polysémie des langues naturelles (un modèle supportant plusieurs interprétations et représentant des systèmes ayant des définitions de type différentes).

De plus, [Bunge, 1972] développe une analyse des concepts de modèle et introduit une distinction entre modèles objet et modèles théoriques.

4.1 Modèles objet

Les modèles objet sont des représentations schématiques de systèmes pour lesquelles une relation de dénotation est établie entre certains éléments du modèle et certaines entités et des propriétés du système représenté.

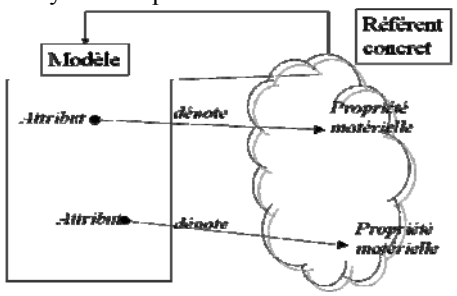

Figure 8. Modèle objet.

Le schéma ci-dessous constitue un modèle objet très sommaire d'une avionique fournissant à l'équipage d'un aéronef sa position verticale. Cette description aurait pu être faite de manière équivalente à l'aide d'une représentation textuelle.

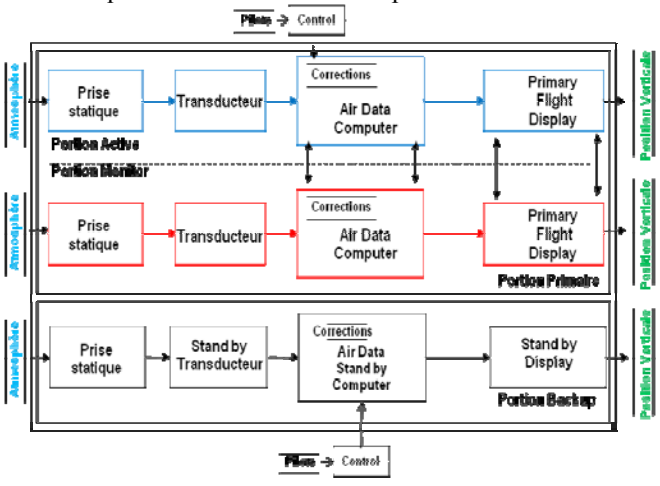

Figure 9. Modèle objet d'un altimètre sensible.

On peut identifier à un certain niveau d'abstraction les éléments constitutifs de ce système : des prises de pression statique, des transducteurs, des calculateurs de données air, des écrans d'affichage et des boitiers de commande (composition du système). On peut également repérer les liaisons entre ces différentes entités et l'organisation du système en voies redondantes (structure ou architecture du système). Enfin on peut identifier l'environnement du système à savoir l'atmosphère et les pilotes. Comme dans tout modèle de nombreux éléments ont été omis.

Malgré les bénéfices que procurent de tels modèles objet, on en perçoit vite les limites. Par exemple, le modèle objet n'indique pas comment des pressions acquises au niveau des prises statiques sont transformées en indication de position verticale de l'aéronef. C'est la raison pour laquelle il est nécessaire de superposer au modèle objet une théorie spécifique

4.2 Modèles théoriques

Les différents attributs des entités du modèle peuvent être liés entre eux dès l'instant où l'on dispose d'une théorie (système abstrait) permettant de décrire l'évolution des propriétés du système représenté les unes vis-à-vis des autres. L'application d'une théorie générale (par exemple, la mécanique des fluides) au cas particulier du modèle considéré (par exemple, hydrostatique de l'atmosphère) constitue une théorie spécifique du système représenté laquelle fournit un mécanisme décrivant plus ou moins exactement le comportement de ce dernier. La figure 10 et l'équation suivantes en sont des illustrations.

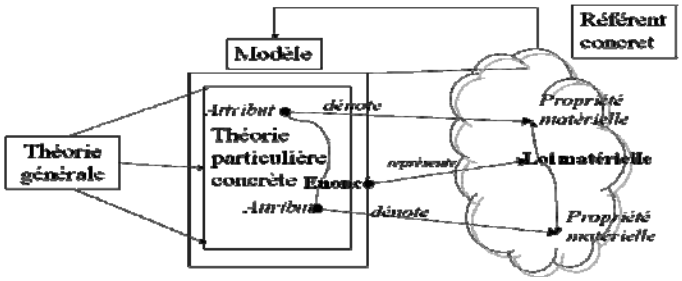

Figure 10. Modèle théorique.

Si l'on considère l'équation suivante :

$$[\text{Eq 1}]: \text{Hp} = \text{f(Ps)} = a * \left[1 - \left[\frac{P_s}{b}\right]^c\right]$$, on observe qu'il s'agit d'un système de symboles, reconnu par toute personne disposant d'une formation élémentaire en mathématiques comme une équation bien formée, établissant l'égalité entre le terme H_p et le terme de droite combinant le symbole P_s et les coefficients a, b et c (non nuls). Si, de plus, le symbole H_p dénote l'altitude par rapport au niveau moyen de la mer et si le symbole P_s dénote la pression statique de l'atmosphère à l'altitude H_p, alors l'équation [Eq 1] désigne la fonction inverse de la fonction $Ps = p(H_p)$ qui est une connaissance (issue d'un corps de connaissances, par exemple, l'atmosphère type internationale (ISA)), laquelle fait référence à une loi physique liant la pression statique à l'altitude dans la troposphère. L'équation [Eq 1] fournit ainsi une image condensée de la relation triangulaire qu'entretiennent entre eux les systèmes symboliques, abstraits et concrets [Bunge, 1974].

4.3 Langages de modélisation

Considérant des langages graphiques tels que [OMG SysML™, 2010], le langage des graphes de liaison - Bond Graphs - [Karnopp, 2006], des langages textuels tels que B [Abrial, 2005], VHDL-AMS [Hervé, 2002] ou encore AADL [SAE-AS5506A, 2009], nous pouvons faire les observations suivantes :

- en premier lieu, ces langages sont tous orientés produit. Ils permettent de réaliser des modèles de systèmes, c'est-à-dire de représenter le résultat d'activités de spécification/conception et non de représenter le processus de spécification/conception, lui-même, qui a conduit à ce système plutôt qu'à un autre. Seuls quelques langages comme par exemple SysML, B ou VHDL-AMS permettent de tracer des éléments de spécification (exigences) à l'aide de clauses particulières (invariants, pré- et post- conditions de B ou assertions de VHDL-AMS) ou des choix de conception (réalisation d'une entité donnée à l'aide d'une architecture particulière en VHDL-AMS).
- en deuxième lieu, les types de systèmes représentables à l'aide de ces langages sont assez différents : les Bond Graphs permettent d'abord de modéliser des systèmes continus transformant différentes formes d'énergie, ce pourquoi AADL n'est pas du tout adapté, ce dernier permettant de ne modéliser que des systèmes discrets. VHDL-AMS revendique le domaine discret, analogique et mixte tandis que SysML s'affirme comme un langage généraliste c'est-à-dire apte à modéliser n'importe quel type de système technologique.
- en troisième lieu, les aptitudes des différents langages, dans leurs domaines respectifs, peuvent être plus ou moins étendues. Ainsi la modélisation d'un système en VHDL-AMS peut permettre non seulement de le représenter mais également de le simuler et enfin de le synthétiser et de le fabriquer. La modélisation d'un système en Bond Graphs permet de le représenter et de le simuler. D'autres enfin, tel SysML, permettent seulement de modéliser le système représenté mais n'autorise pas sans autres frais sa simulation et encore moins sa production.
- en dernier lieu, ces langages comportent essentiellement deux types de traits : des déclarations et des instructions. Les déclarations permettent d'introduire des (types d') objets ou des (types d') attributs d'objet du modèle et notamment les objets et attributs qui permettent de dénoter composants et propriétés des composants du système représenté. Les instructions, quant à elles, permettent de définir l'évolution de l'état du modèle qui représente plus ou moins exactement l'évolution du système représenté en fonction du temps ou d'autres paramètres. Ainsi les déclarations permettent de définir le modèle objet, tandis que les instructions permettent de définir la théorie spécifique associée au modèle objet pour former le modèle théorique du système référent.

5 LA MODELISATION DES EXIGENCES

Le problème posé par les langages de modélisation est le suivant : ils ne rendent en général pas compte des moments charnière du processus de conception, tracent peu ou pas les résultats de ce processus, en particulier la spécification des exigences et leur ingénierie pendant tout le processus de conception est ignorée de la plupart des langages de modélisation à l'exception notable de SysML et de ses diagrammes d'exigences.

5.1 Exigences textuelles et exigences basées propriétés

La spécification d'un système est constituée d'ensembles d'obligations et d'interdictions relatives à sa structure et à son comportement. De tels énoncés sont sous le contrôle des clients, des autorités de régulation ou d'autres parties prenantes et le concepteur du système est tenu de les respecter. Parmi ces énoncés, on peut introduire une distinction entre d'une part la notion d'attente ou exigence textuelle (expression textuelle

plus ou moins précise d'une demande d'une partie prenante) et d'autre part la notion d'exigence basée propriété ou PBR.
Dans [Micouin, 2008], nous avons proposé une théorie des PBRs qui, au contraire des exigences textuelles (-TBR-), sont aptes à s'intégrer dans une approche d'ingénierie basée sur des modèles. Selon nous, les PBRs appartiennent au même paradigme que le MBSE tandis que les TBRs appartiennent à celui de l'ingénierie basée documents. C'est, selon nous, la raison principale de l'inadéquation des diagrammes d'exigences de SysML au MBSE.
Si la thématique exigence/propriété n'est pas d'aujourd'hui, [Kamsu-Foguem, 2006], notre approche en construit la problématique sur la base de la théorie des propriétés exposée dans [Bunge, 1977]. Nous définissons une PBR comme une contrainte posée sur une propriété que possède un objet ou un ensemble d'objets lorsqu'une condition est satisfaite. Cette définition peut s'exprimer formellement de la manière suivante : when C $\Rightarrow$ val(O.P) $\subseteq$ D ce qui signifie : lorsque la condition C est réalisée, la propriété P de l'objet O doit se situer dans le domaine D (l'interdiction s'en déduit immédiatement). Aucune hypothèse n'est faite sur l'ensemble D, lequel peut être aussi compliqué qu'on le souhaite. Par exemple, D peut-être une fonction comme ici D={(P_s, H_p)/ $P_s \in [P_{min}, P_{max}]$ et H_p =f(Ps)} où H_p =f(Ps) correspond à l'équation [Eq 1] décrite ci-dessus au § 4.2.
Comme déjà vu, les propriétés d'un objet concret peuvent être classées en deux catégories : structurelles d'une part et comportementales d'autre part. Conséquence de cette typologie des propriétés, il existe au moins deux types d'exigences, les exigences structurelles et les exigences comportementales. S'agissant, par exemple, d'un système embarqué fournissant les données air (ADC pour Air Data Computer) d'un aéronef, on peut poser sur lui des exigences structurelles de masse, d'encombrement, de forme, de redondance, de dissimilarité, etc. et des exigences comportementales concernant le domaine de définition, la précision, le seuil de sensibilité des données fournies, la réponse à des signaux d'entrée caractéristiques.
Le processus d'élicitation d'une exigence est un processus permettant de passer d'une exigence textuelle à une ou plusieurs exigences bien formées. Nous faisons l'hypothèse que cette élicitation est en pratique toujours possible, même si elle peut s'avérer très compliquée. L'effort considérable de standardisation aéronautique réalisé par les autorités de régulation, les professionnels et les organisations (SAE, ARINC, RTCA, EUROCAE, etc.) est un témoignage de ce travail d'élicitation d'exigences textuelles conduisant en pratique à l'expression d'exigences bien formées.

Altitude Feet	Altitude Meters	Tolerance ± Feet	Tolerance ± Meters
0.	0.	25.	8.
1000.	305.	25.	8.
2000.	610.	25.	8.
3000.	914.	25.	8.
4000.	1219.	25.	8.
5000.	1524.	25.	8.
8000.	2438.	30.	9.
11000.	3353.	35.	11.
14000.	4267.	40.	12.
17000.	5182.	45.	14.
20000.	6096.	50.	15.
30000.	9144.	75.	23.
40000.	12192.	100.	30.
50000.	15240.	125.	38.

Tableau 1. Tolérances sur l'altitude exigées par l'AS8002A

Par exemple, Si l'on considère dans la réglementation canadienne, l'exigence suivante concernant les hélicoptères de transport :
529.1303 Les instruments de vol et de navigation suivants sont exigés : (b) un altimètre sensible;
Celle-ci a fait l'objet d'un travail d'élicitation [AS8002A, 1996] qui donne notamment la performance minimum d'une indication altitude fournie par un calculateur de données air (voir tableau 1, ci-dessus).
Plus encore, en introduisant l'ensemble $SAT_K(Ex)$ des objets de type K qui satisfont l'exigence Ex, on peut définir deux relations entre exigences: une relation d'ordre partielle « *être plus contraignante que*» et une opération de conjonction entre exigences.

1. D'une part, une exigence Ex-1 *est plus contraignante* qu'une exigence Ex-2 si et seulement si l'ensemble SAT_K (Ex-1) est un sous ensemble SAT_K (Ex-2) i.e. Ex-1$\geq$Ex-2$\Leftrightarrow SAT_K$ (Ex-1)$\subseteq SAT_K$ (Ex-2).
2. D'autre part, l'exigence Ex *est la conjonction* de Ex-1 et Ex-2 si et seulement si SAT_K (Ex) = SAT_K (Ex-1) $\cap$ SAT_K (Ex-2) et on note alors Ex=Ex-1$\wedge$Ex-2.

Mathématiquement parlant, l'ensemble des exigences portées par un système génère une structure de semi-treillis pour laquelle la relation « être plus contraignante que» et l'opérateur de conjonction entre exigences se déduisent l'un de l'autre.
Dans le contexte de cette théorie, la dérivation d'une exigence est une transformation qui substitue à une exigence de niveau système un ensemble d'exigences de niveau sous systèmes moyennant un ensemble d'hypothèses de conception. La dérivation est donc une opération conditionnelle qui ne vaut que si les hypothèses restent valides. Formellement, la relation CC $\Rightarrow$ Ex $\leq$ Ex1$\wedge$ ..$\wedge$Exn établit que si CC est réalisée alors la conjonction des exigences dérivées est plus contraignante que l'exigence dont elles dérivent, ou encore que la satisfaction des exigences dérivées est une condition suffisante (mais non nécessaire) à la satisfaction de l'exigence « mère ». Il n'y a pas lieu de s'étonner de ce que cette condition soit suffisante mais non nécessaire, dans le mesure où les problèmes de conception de systèmes sont des problèmes dits « inverses », c'est-à-dire, pour lesquels il existe a priori une multiplicité de solutions, chacune d'elles étant suffisante pour résoudre le problème mais ne s'imposant pas nécessairement par rapport aux autres.

5.2 Représenter les exigences dans les modèles

La façon la plus simple de prendre en compte une PBR dans un modèle de système est d'utiliser des assertions telles que les proposent des langages comme le B, VHDL-AMS ou d'autres langages encore.
Cela se justifie par le fait que l'on sait que formellement parlant l'expression [when P => Q] est équivalente à l'expression booléenne [Q or not(P)].
En VHDL-AMS, la syntaxe d'une assertion est donnée par :

```
[lab:] assert cond_bool [..] ;
```

On peut donc écrire

```
Ex1: assert Tol(0,5000,25,HpI,Hpr);
```

qui permet d'exprimer l'exigence suivante :
Ex1 : when AC.Altitude $\in$[0ft,5000ft[$\Rightarrow$ $|HpI-Hpr| \leq 25ft$
autrement dit quand l'entité Aeronef est à une altitude située entre 0 et 5000ft, l'erreur tolérée sur l'indication de position verticale (par rapport à une référence Hpr, fournie, par exemple, par une installation d'essais en vol) doit être inférieure à 25ft.
De même, on exprimer l'exigence :

```
Ex2: assert Tol(5000,11000,35,HpI,Hpr);
```

Et ainsi de suite pour chaque palier du tableau 1.

La juxtaposition de plusieurs assertions Ex1 .. Exn dans la partie instructions parallèles de la déclaration de l'entité correspond exactement ce que nous avons appelé la conjonction Ex1 ∧.. ∧Exn de ces exigences.
Mais, nous aurions pu également exprimer cette conjonction d'exigences de manière plus synthétique en écrivant

```
Ex: assert Tol(Table_Alt,HpI,Hpr);
```

où Table_Alt représente le tableau 1 ci-dessus.
La fonction Tol est ici une fonction booléenne qui retourne V si l'erreur | HpI-Hpr| est inférieure à la tolérance fournie dans le tableau selon le niveau auquel l'aéronef évolue.
On peut donc modéliser l'ensemble des exigences portées par une entité par une liste d'assertions située dans la partie instructions parallèles de la déclaration d'une entité.
Voici la liste des exigences que nous allouons à l'entité Avionique (exigences du système selon l'EIA 632).

```
entity Avionique is .. ;
   function Tol_Hi (..) return boolean ;
   function P_AV (..) return Poids ;
   function Redondance_AV(..) return ..;
begin
   Ex_AV1: assert Tol_Hi (Tableau_Alt, HpI, Hpr);
   Ex_AV2: assert P_AV ≤ 15;
   Ex_AV3: assert Redondance_AV≥3;
end Avionique;
```

Pour l'exigence Ex_AV2, dans l'expression P_AV ≤ 15, la fonction P_AV retourne le poids de l'avionique AV, lequel doit être inférieur à 15kg (budget de poids).
Pour l'exigence Ex_AV3, la fonction entière Redondance_AV retourne le nombre de redondances de l'entité AV, lequel doit être supérieur ou égal à 3.

5.3 Concevoir l'architecture fonctionnelle d'un système

La conception fonctionnelle d'une entité, telle que celle présentée ci-dessus au §3.1 figure 6, consiste à envisager un certain nombre d'architectures fonctionnelles alternatives et d'en retenir une qui serait l'architecture fonctionnelle préférée en lui allouant les exigences comportementales du système. Nous en proposons ci-dessous un exemple qui illustre la façon dont certaines exigences sont prises en compte et dérivées :

```
architecture Fonctionnelle of Avionique is
   function TolAA (..) return boolean ;
   function TolCA (..) return boolean ;
   function TolDA (..) return boolean ;
   function TolAB (..) return boolean ;
   .. ;
   function P_AV (..) return Poids is .. end ;
   function Tol_Hi (..) return Boolean is .. end ;
   function Redondance_AV (..) return Nbr_Redondance
       is begin .. end ;
begin
   voie_A    :process ..
   begin
       Psm <= Acq_Pression_Statique(Ps);
       Ex_VAn : assert TolAA(..);
       Hp <= Cal_Alt_Baro(Psm);
       Ex_VAn+1: assert TolCA (..);
       Afficher_Position_Vertical(Hp);
       Ex_VAn+2: assert TolDA (..);
   end process ;
   voie_B    :process .. end process ;
   voie_secours: process .. end process ;
end Fonctionnelle;
```

Cette architecture fonctionnelle dévoile la présence de trois processus concurrents (voie A, B et secours) qui assurent une redondance fonctionnelle satisfaisant au niveau fonctionnel l'exigence Ex_AV3 portée par l'entité Avionique.
Par ailleurs, le processus Voie_A est composé de trois activités réalisées séquentiellement. Chacune de ces activités est ici suivie d'une assertion (exigence) exprimant une contrainte sur la ou les sorties de l'activité (on suppose que la voie B a fait l'objet d'une conception identique tandis que celle de la voie Secours est différente pour des raisons de dissimilarité).
La première exigence Ex_VAn spécifie que l'erreur maximum sur la mesure de la pression statique par rapport à la pression statique réelle à l'altitude de référence Hpr doit rester tolérable. La deuxième, Ex_VAn+1 spécifie une tolérance sur l'erreur maximum sur le calcul de l'altitude, tandis que la troisième Ex_VAn+2 spécifie une tolérance sur l'erreur maximum d'affichage de l'altitude.
Le fait que ces exigences soient des exigences dérivées de l'exigence Ex_AV1 est matérialisé par le corps de la fonction Tol_Hi (déclaré dans la partie déclarative de l'architecture) de la manière suivante :

```
       function Tol_Hi (..)
        return Boolean is ;
       begin  return
       TolAA(..)and TolCA(..) and TolDA(..) and
       TolAB(..)and TolCB(..) and TolDB(..) and
       TolAS(..)and TolCS(..) and TolDS(..);
end;
```

Le corps de cette fonction Tol_Hi décrit la façon dont l'exigence Ex_AV1 est dérivée et allouée aux différents éléments fonctionnels de l'architecture de l'entité Avionique. Si cette dérivation a été correctement faite alors elle assure que si Ex_VAn, Ex_VBn, Ex_VSn, Ex_VAn+1, etc. sont satisfaites alors Ex_AV1 est également satisfaite (condition suffisante mais non nécessaire). Ainsi la fonction Tol_Hi qui figure dans l'exigence Ex_AV1 portée par l'entité Avionique est la fonction d'allocation de cette exigence, elle doit être définie pour toute architecture fonctionnelle de l'entité.

5.4 Concevoir l'architecture physique d'un système

Si maintenant on considère une architecture physique de l'entité Avionique, telle que celle présentée ci-dessus au § 3.1 figure 7, celle-ci se distingue d'une architecture fonctionnelle correspondante par son aptitude à satisfaire en plus des exigences comportementales les exigences structurelles assignées à l'entité.
Ci-dessous l'architecture physique de l'avionique:

```
architecture Physique of Avionique is .. ;
   function P_AV (..) return Poids is ..;
   function Tol_Hi (..) return Boolean is ..;
   function Redondance_AV (..) return Nbr_Redondance
       is begin .. end ;
begin
   Va : Voie_Primaire port map (..) ;
   Vb : Voie_Primaire port map (..) ;
   Vs : Voie_Secours port map (..) ;
 end Physique;
```

Elle s'appuie sur trois instances de composants Voie dont deux sont des voies primaires tandis que la dernière est une voie secours. Ces voies sont connectées aux ports de l'avionique par les connexions adéquates (port map). Ces instances assurent une redondance satisfaisant au niveau physique l'exigence Ex_AV3 portée par l'entité Avionique.
Par ailleurs, la fonction P_AV ainsi définie :

```
function P_AV return Poids is ;
begin
  return 2*P_VP + P_VS;
end;
```

assure que l'exigence Ex_AV2 sera satisfaite si les exigences VP_2 et VS_2 sont satisfaites, VP_2 et VS_2 étant des exigences respectivement portées par les entités Voie_Primaire et Voie_Secours, déclarées ci-dessous.

```
entity Voie_Primaire  is ..;
begin
Ex_VP_1: assert Tol_alt(HpI,AC.Altitude);
Ex_VP_2: assert P_VP ≤ 6;
end Voie_Primaire ;et
```

```
entity Voie_Secours  is ..;
begin
Ex_VS_1: assert Tol_alt(HpI,AC.Altitude);
Ex_VS_2: assert P_VS ≤ 3;
end Voie_Secours ;
```

La dérivation des exigences Ex_VP1 et Ex_VS1 sont analogues à celle de l'exigence Ex_AV1, ci-dessus.

5.5 Poursuivre le processus de conception d'un système

Chaque entité Voie_Primaire et Voie_Secours fait l'objet d'une conception fonctionnelle qui est ici quasi-immédiate, puisqu'il s'agit d'allouer un processus de l'architecture fonctionnelle de l'avionique à cette architecture. Elle fait ensuite l'objet d'une architecture physique.

Ces deux architectures sont ébauchées ci-dessous. L'architecture fonctionnelle met en évidence la ou les chaînes fonctionnelles que l'entité réalise et les exigences associées aux éléments de ces chaînes tandis que l'architecture physique met en évidence les composants (entités composantes) de l'entité et les connexions entre ces entités (endostructure).

On a ainsi :

```
architecture Fonctionnelle of Voie_Primaire is
   begin
        Psm <= Acq_Pression_Statique(Ps);
        Ex_VPn : assert TolP(..);
        Hp <= Cal_Alt_Baro(Psm);
        Ex_VPn+1: assert TolH (..);
        Afficher_Position_Vertical(Hp);
        Ex_VPn+2: assert TolD (..);
   end process ;
end Fonctionnelle;
```

pour laquelle, l'activité Acq_Pression_Statique fait l'objet de la décomposition suivante :

```
function Acq_Pression_Statique (..)return .. is
begin
  Ps == CapterPression;
  Psm <= ConvertirAN(Ps);
  return Psm ;
end;
```

et:

```
architecture Physique of Voie_Primaire is .. ;
begin  S : SondePression port map (..);
       T : Transducteur port map (..);
       ADC : CalcDonneesAir port map (..);
       PFD : SystemeAffichage port map (..);
end Physique;
```

Cette architecture physique fait référence à des entités déclarées par ailleurs telles que SondePression, Transducteur, CalcDonneesAir etc., lesquelles sont elles mêmes porteuses d'exigences comme dans le cas du calculateur de données air :

```
entity CalcDonneesAir is ..;
begin
Ex_ADC_1: assert assert Tol (Hp, Psm);
Ex_ADC_2: assert Poids_ADC ≤ 2.5;
end CalcDonneesAir;
```

A cette dernière entité, on va associer une architecture dont le rôle sera de réaliser l'équation [Eq 1] (§ 4.2).

Le processus de conception s'arrête lorsque les différentes entités de plus bas niveau sont soit directement réalisables (fabricables ou codages) soit achetables. La réalisation commence avec la production ou l'acquisition de ces entités de plus bas niveau, leur vérification unitaire par rapport aux exigences qui leur sont assignées, puis leur intégration et la vérification des entités intégrées par rapport aux exigences assignées à ces entités intégrées.

6 CONCLUSION

Dans cet article, nous pensons avoir apporté une contribution à l'effort de définition d'une ingénierie des systèmes basée sur les modèles. Nous avons formulé ce que nous pouvons entendre par modèles et montré la diversité des langages de modélisation tant du point de vue de l'extension des systèmes concrets pour lesquels ils sont aptes à produire des modèles que des performances des modèles qu'ils permettent de définir. Nous avons ensuite rappelé les principaux éléments d'une théorie des PBRs, capable de s'inscrire dans une approche MBSE alors que les exigences textuelles nous apparaissent comme antagoniques à cette approche. Nous avons ensuite montré comment il est possible (1) de définir, dans un langage de modélisation particulier, des PBRs au sein même d'un modèle de conception de système, (2) d'appliquer à ce modèle en développement un processus de conception système conforme à l'état de l'art et (3) de dériver les exigences initiales du système pour les allouer progressivement aux différentes entités du modèle. Ce faisant, nous pensons proposer des processus de définition des PBRs et de définition des solutions fonctionnelle et physique des systèmes qui renouvellent les pratiques actuelles du MBSE et les rendent plus aptes au développement des systèmes d'aujourd'hui et de demain.

Toutefois, de nombreux champs d'investigation restent à explorer notamment celui de l'articulation des modèles d'exigences et de conception fonctionnelle et physique tels que nous les avons évoqués ci-dessus avec les modèles de sûreté (arbres de défaillance, diagrammes de fiabilité associés aux analyses des dangers et de sûreté) que nous avions identifié dans [Micouin, 2009] et pour lesquels l'intégration dans une approche MBSE reste à faire.

7 REFERENCES

Abrial J.R, The B-Book: Assigning Programs to Meanings, Cambridge University Press, Nov 2005.

Hervé Y, *VHDL-AMS : Applications et enjeux industriels*, Dunod, 2002.

Bunge M.A., Concepts of model in *Method, Model and Matter*, Reidel Publishing Company (RPC), 1972.

Bunge M.A., Semantics I: *Sense and Reference*, Treatise on Basic Philosophy (TBP), vol 1, , RPC, 1974.

Bunge M.A., Ontology I: *The furniture of the world,* TBP, vol 3, RPC, 1977.

Bunge M.A., Ontology II: *A world of systems*, TBP, vol 4, RPC,1979.

EIA 632, *Processes for Engineering a System,* GEIA, 2003.

INCOSE Systems Engineering Vision 2020, Document INCOSE-TP-2004-004-02, Version 2.03, Sep 2007.

ISO/IEC 15288-2002, *Life-Cycle Management—System Life Cycle Processes,* Octobre 2002.

Kamsu-Foguem B. and Chapurlat V., *Requirements modelling and formal analysis using graph operations,* International Journal of Production Research, vol. 4 (n° 17) 2006

Karnopp D.C, Margolis, D.L, Rosenberg, R.C, System Dynamics: Modeling and Simulation of Mechatronics Systems, Wiley, Jan 2006.

Luzeaux D , Ruault J.R, (2008) Systèmes de systèmes vol 1 : Concepts et Illustrations, Hermes.

Micouin P, (2008) *Toward a property based requirements theory: System requirements structured as a semilattice*, Journal of Systems Engineering, vol 11, Issue 3 , Wiley.

Micouin P.M, (2009) Ingénierie des exigences et conception des systems d'aéronefs, 5ième conference de l'AFIS, Paris, sep 2009.

Systems Modeling Language (OMG SysML^{TM}), Ver *1.2*, 2010.

SAE-AS5506A, Architecture Analysis & Design Language (AADL), Jan 2009.

SAE-AS8002A Air Data Computer : Minimum Performance Standard, Aerospace Standard SAE International, 1996-09.